Data Analysis and Visualization Exercise 10

Matthias Heinig, Felix Brechtmann, Daniela Klaproth-Andrade, Julien Gagneur

Section 00 - Getting Ready

1. Make sure you have already installed and loaded the following libraries:

```
library(ggplot2)
library(data.table)
library(magrittr)
library(tidyr)
library(dplyr)
library(patchwork) # optional, makes plots nicer
```

Section 01 - Linear regression for Predicting Heights

To start, read the provided heights dataset using the following line of code (it's your own heights data):

```
heights <- fread("extdata/height.csv") %>% na.omit() %>%
  .[, sex:=as.factor(toupper(sex))]
heights
```

1. Predict each student's hight, given their sex and their parents heights.

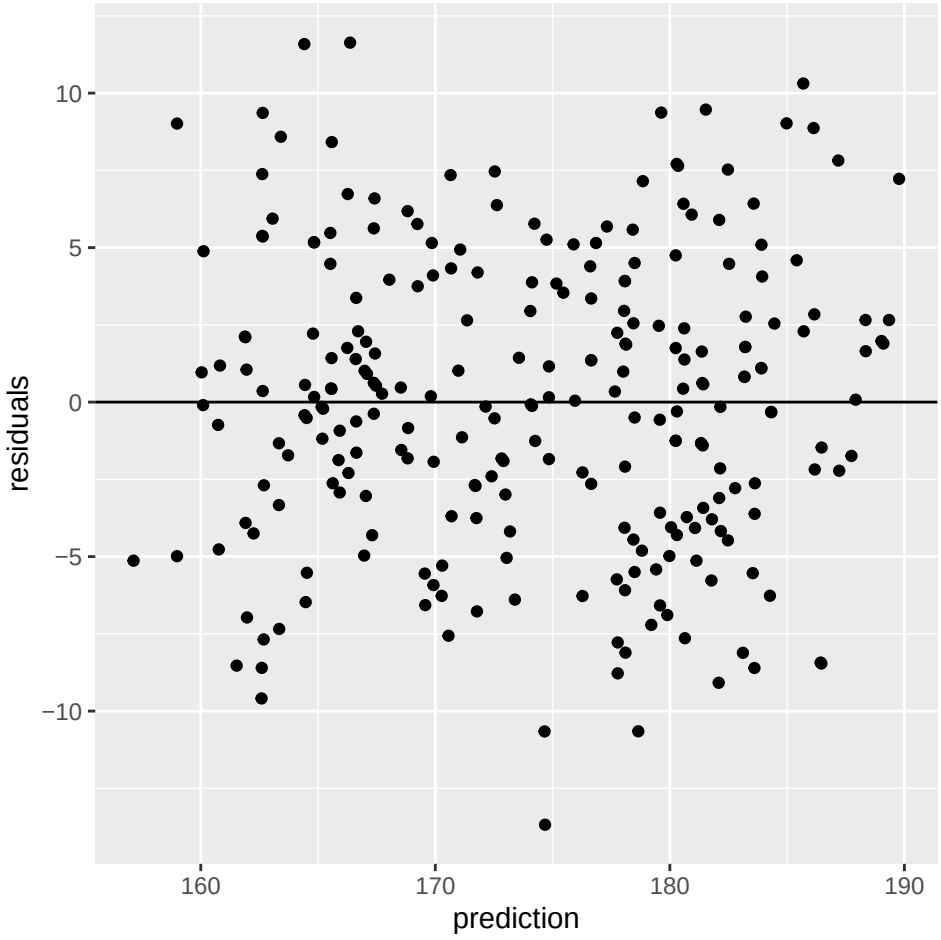
```
m <- heights[, lm(height ~ sex + mother + father)]
summary(m)
```

```
##
## Call:
## lm(formula = height ~ sex + mother + father)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.6777  -3.5802   0.1564   3.3738  11.6338
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.07027    8.77308   4.795 2.80e-06 ***
## sexM        14.00534    0.60830  23.024  < 2e-16 ***
## mother       0.37054    0.04560   8.126 2.08e-14 ***
## father       0.36050    0.03869   9.316  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.82 on 249 degrees of freedom
```

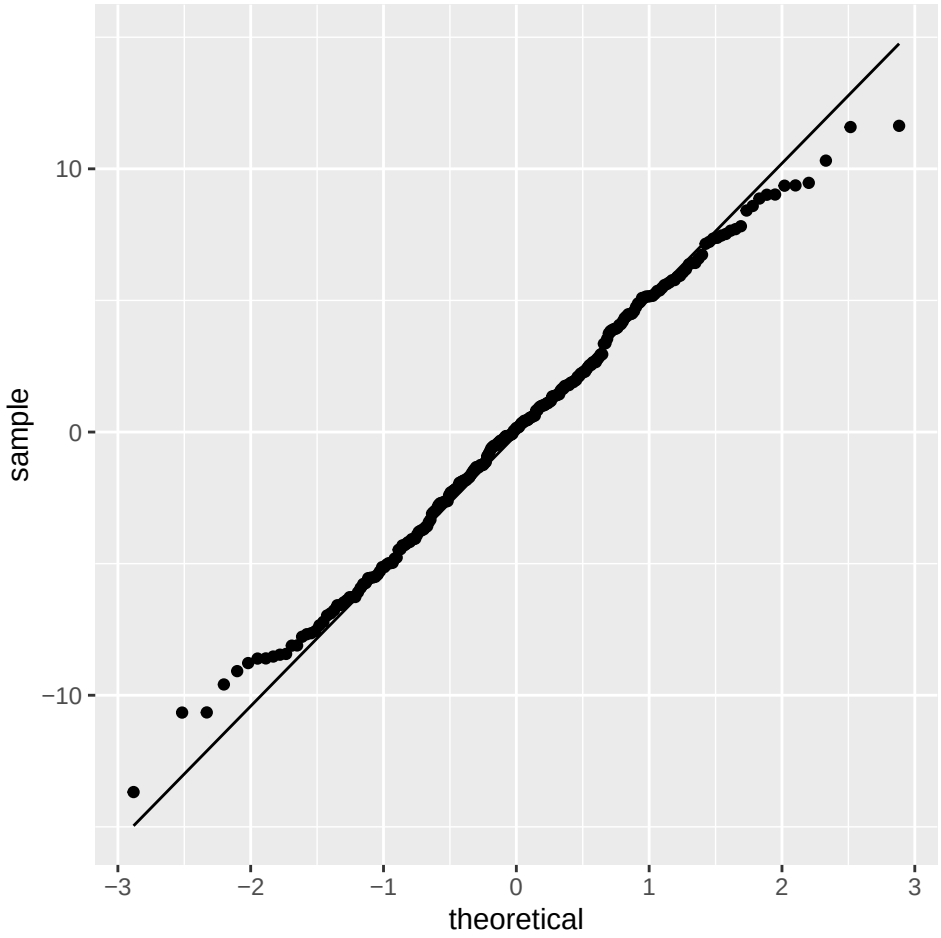
```
## Multiple R-squared:  0.7369, Adjusted R-squared:  0.7337
## F-statistic: 232.5 on 3 and 249 DF,  p-value: < 2.2e-16
```

2. Check the plot of the residual vs the predicted values and the Q-Q plot of the residuals. Do these plots provide evidence against the assumptions of linear regression?

```
prediction = data.table(prediction = predict(m), residuals = residuals(m))
ggplot(prediction, aes(prediction, residuals)) + geom_point() + geom_hline(yintercept = 0)
```




```
ggplot(prediction, aes(sample = residuals)) + geom_qq() + geom_qq_line()
```



3. Let's focus on one sex. Fit a linear model for each male's height given the father's height. Then, fit another linear model for the father's height given each male's height. To do so, subset the table to only include male students.

```
heights_male <- heights[sex == 'M']

m1 <- heights_male[, lm(height ~ father)]
m1
```

```
##
## Call:
## lm(formula = height ~ father)
##
## Coefficients:
## (Intercept)       father
##     92.1599       0.5015
```

```
m2 <- heights_male[, lm(father ~ height)]
m2
```

```
##
## Call:
```

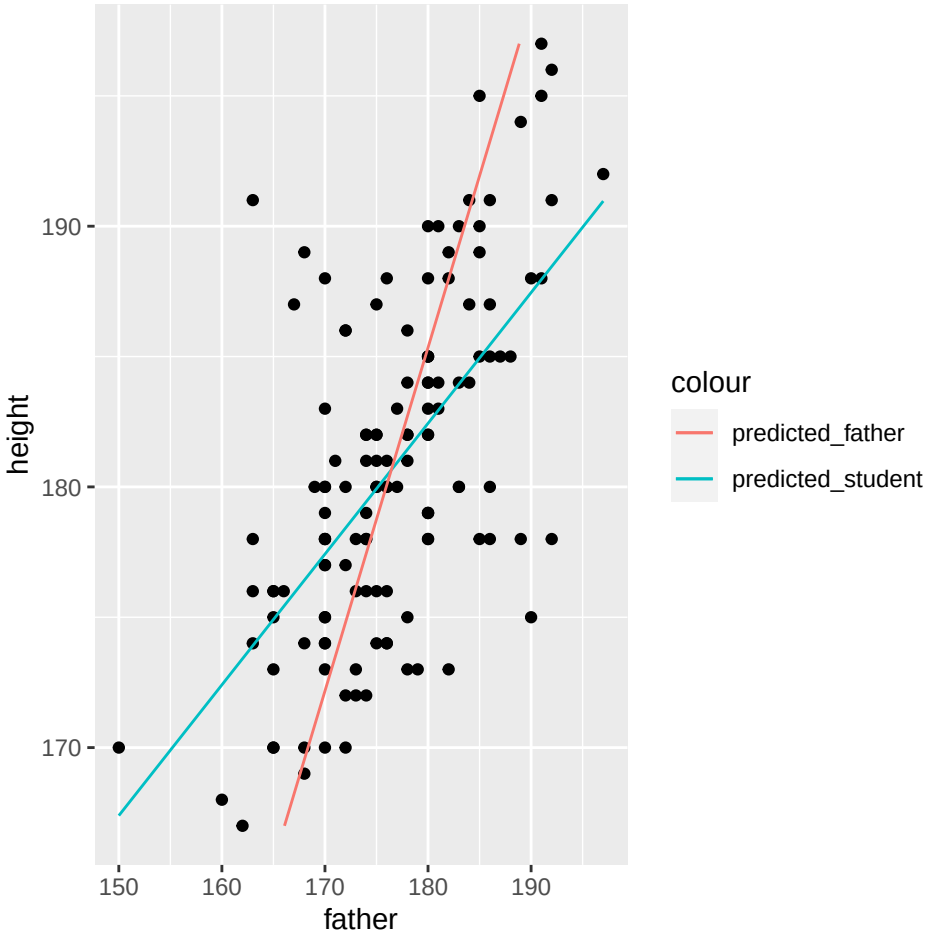
```
## lm(formula = father ~ height)
##
## Coefficients:
## (Intercept)       height
##     39.2836       0.7592
```

4. Predict each student's height given the father's height (`predict()`) and predict each father's height given each student's height. Store both predictions into new columns of a data table. Then, plot the original data and additionally both regression lines.

```
heights_male[, predicted_student_height := predict(m1) ]
heights_male[, predicted_father_height := predict(m2)]
head(heights_male)
```

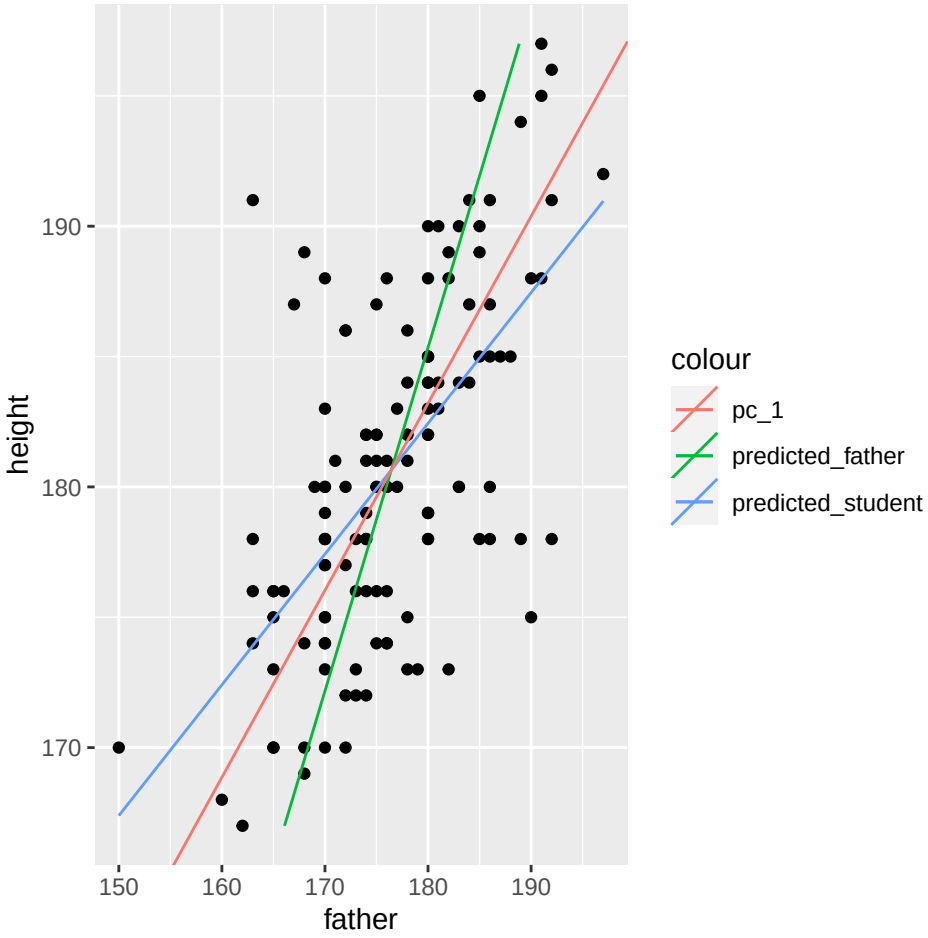
```
##    height sex mother father predicted_student_height predicted_father_height
## 1:    197   M    175    191                 187.9537                188.8492
## 2:    196   M    163    192                 188.4552                188.0900
## 3:    195   M    171    185                 184.9445                187.3307
## 4:    195   M    168    191                 187.9537                187.3307
## 5:    194   M    164    189                 186.9506                186.5715
## 6:    192   M    168    197                 190.9629                185.0531
```

```
ggplot(heights_male) +
  geom_point(aes(father, height)) +
  geom_line(aes(father, predicted_student_height, color = 'predicted_student')) +
  geom_line(aes(predicted_father_height, height, color = 'predicted_father'))
```



5. Additionally, run a PCA on the same subset of the data and plot the first principal component into the same figure.

```
pca_obj <- princomp(heights_male[, .(height, father)])
slope <- pca_obj$loadings["height","Comp.1"] / pca_obj$loadings["father","Comp.1"]
intercept <- pca_obj$center['height'] - pca_obj$center['father'] * slope

ggplot(heights_male) +
  geom_point(aes(father, height)) +
  geom_line(aes(father, predicted_student_height, color = 'predicted_student')) +
  geom_line(aes(predicted_father_height, height, color = 'predicted_father')) +
  geom_abline(aes(intercept = intercept, slope = slope, color = 'pc_1'))
```



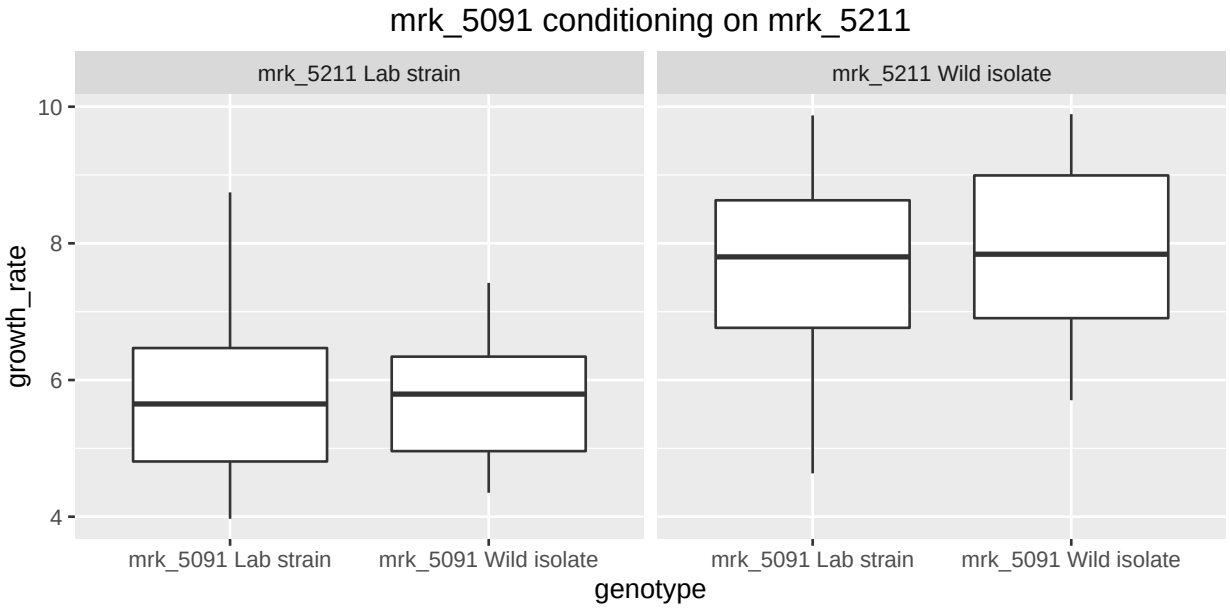
6. Interpret the plot from above. How can we explain the different slopes of the two linear models and the pca?

```
# The linear models minimizes the RSS along the predicted coordinate.
# In this specific case the height for the first model of the students heights and the
# fathers height for the second model predicting the fathers heights.
#
# PCA minimizes the sum of squares (along both coordinates). And hence, the
# distance perpendicular to the first principal component.
```

Section 02 - Adjusting for confounding variables

Yeast QTL

Recall the yeast QTL data set from the previous exercises and lecture. In particular, we consider once again the question: Does marker 5091 still associate with growth in maltose when conditioned on marker 5211? Here is the plot of the data:




```
growth <- fread(file.path(eqtl_dir, "growth.txt"))
growth <- growth %>% melt(id.vars="strain", variable.name='media', value.name='growth_rate')
growth <- growth[media=="YPMalt"]

genotype <- fread(file.path(eqtl_dir, "genotype.txt"))
genotype <- genotype[, .(strain, mrk_5211, mrk_5091)]

head(genotype)
```

```
##     strain     mrk_5211     mrk_5091
## 1: seg_01B   Lab strain   Lab strain
## 2: seg_01C   Lab strain   Lab strain
## 3: seg_01D Wild isolate Wild isolate
## 4: seg_02B   Lab strain Wild isolate
## 5: seg_02C   Lab strain   Lab strain
## 6: seg_02D Wild isolate   Lab strain
```

```
head(growth)
```

```
##     strain  media growth_rate
## 1: seg_01B YPMalt    6.720447
## 2: seg_01C YPMalt    7.429273
## 3: seg_01D YPMalt    6.905589
## 4: seg_02B YPMalt    4.924324
## 5: seg_02C YPMalt    4.413402
## 6: seg_02D YPMalt    7.926200
```

1. Run a linear model predicting the growth given both genotypes and interpret the result. Call this model `full`.

```r
table <- merge(growth, genotype)
full <- table[, lm(growth_rate ~ mrk_5211 + mrk_5091)]
summary(full)
```

```
##
## Call:
## lm(formula = growth_rate ~ mrk_5211 + mrk_5091)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.04628 -0.93685  0.02645  0.84925  3.03939
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             5.7064     0.1509  37.809   <2e-16 ***
## mrk_5211Wild isolate    1.9743     0.2008   9.834   <2e-16 ***
## mrk_5091Wild isolate    0.1747     0.2006   0.871    0.385
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.171 on 151 degrees of freedom
##   (4 observations deleted due to missingness)
## Multiple R-squared:  0.4357, Adjusted R-squared:  0.4282
## F-statistic: 58.29 on 2 and 151 DF,  p-value: < 2.2e-16
```

```r
# We can see, that the p-value for the beta of the mrk_5091 is high.
# Hence, we cannot reject the null hypothesis that the beta for this marker is different from 0.
```

2. Create a reduced model that only depends on the genotype of `mrk_5211`. Then run anova to compare the full and the reduced model. What do you conclude?

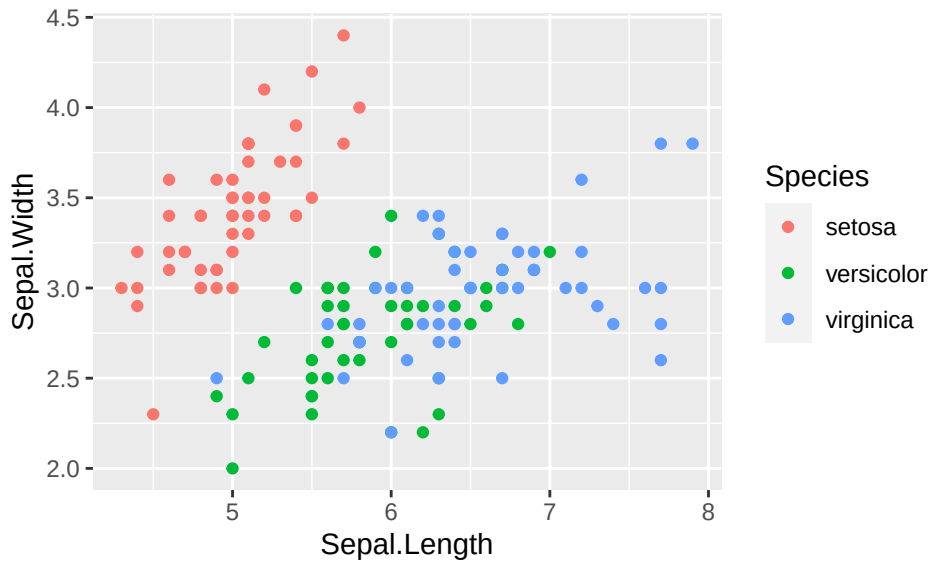
```r
reduced <- table[, lm(growth_rate ~ mrk_5211)]

# Ho: adding the genotype information of mrk_5091 does not improve the model.

## note: observe the probability under the null hypothesis of an F statistic as
## extreme as the one observed here is rather high (> 0.05) so we would not
## reject the null hypothesis.

anova(reduced, full)
```

```
## Analysis of Variance Table
##
## Model 1: growth_rate ~ mrk_5211
## Model 2: growth_rate ~ mrk_5211 + mrk_5091
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    152 208.28
## 2    151 207.23  1    1.0414 0.7588 0.3851
```

Iris dataset

Recall the plot showing the relationship between sepal width and sepal length in the Iris dataset:

```
library(datasets)
library(ggplot2)
data(iris)
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species)) +
  geom_point()
```



1. Fit three linear models predicting Sepal.Width from Sepal.Length: the base model that simply predicts sepal width from sepal length, one where you use the species as a covariate in linear regression (i.e., different intercept for different species) and one where you use separate slopes and intercepts for different species (Hint: use the `*` operator in `lm`: `lm(y ~ x1 * x2)`)

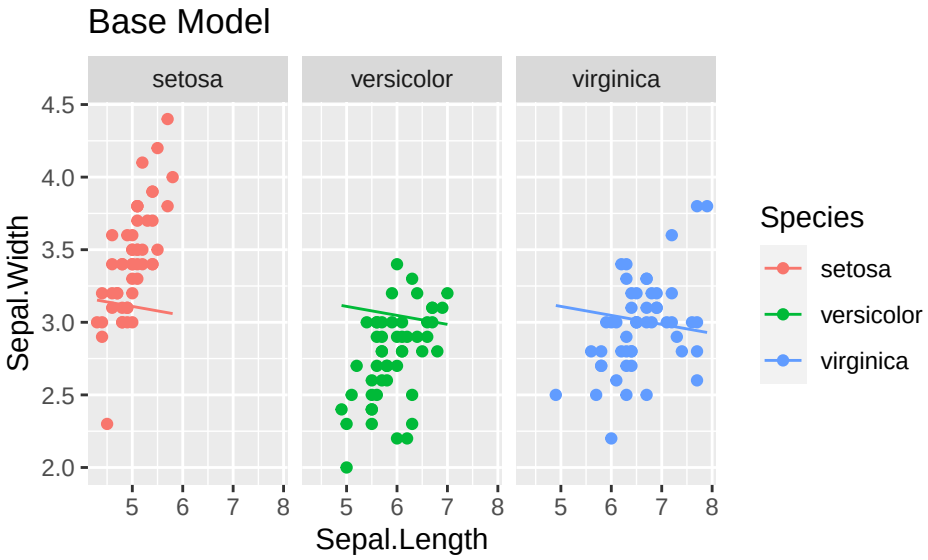
2. Overlay the resulting fits on the plot above (Hint: use `predict` to generate predictions)

3. Use `anova` to test if the second model is a better model than the base and also if the third model is better than the second.

```
# Different intercepts
base_model <- lm(Sepal.Width ~ Sepal.Length, data=iris)
base_model
```

```
##
## Call:
## lm(formula = Sepal.Width ~ Sepal.Length, data = iris)
##
## Coefficients:
##  (Intercept)  Sepal.Length
##      3.41895      -0.06188
```

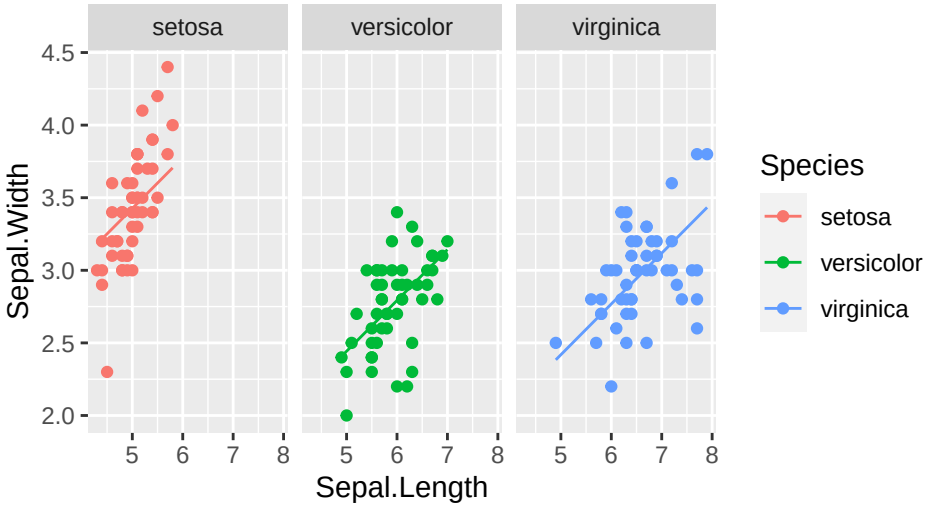
```
iris$base_preds <- predict(base_model, newdata=iris)
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species)) +
  geom_point() +
  geom_line(aes(x=Sepal.Length, y=base_preds, color=Species)) +
  facet_grid(. ~ Species) + ggtitle("Base Model")
```




```
model_species_intercept <- lm(Sepal.Width ~ Sepal.Length + Species, data=iris)
model_species_intercept
```

```
##
## Call:
## lm(formula = Sepal.Width ~ Sepal.Length + Species, data = iris)
##
## Coefficients:
##       (Intercept)       Sepal.Length  Speciesversicolor   Speciesvirginica
##            1.6765             0.3499            -0.9834            -1.0075
```

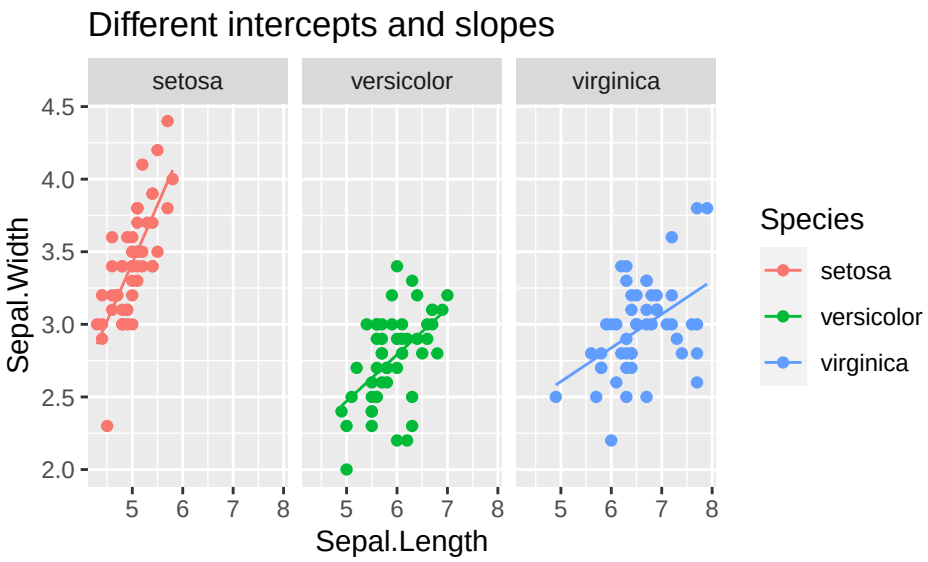
```
iris$preds <- predict(model_species_intercept, newdata=iris)
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species)) +
  geom_point() +
  geom_line(aes(x=Sepal.Length, y=preds, color=Species)) +
  facet_grid(. ~ Species) + ggtitle("Different intercepts, same slope")
```



```r
# Different slopes and intercepts per species
model_species_intercept_slope <- lm(Sepal.Width ~ Sepal.Length*Species, data=iris)
model_species_intercept_slope
```

```
##
## Call:
## lm(formula = Sepal.Width ~ Sepal.Length * Species, data = iris)
##
## Coefficients:
##                       (Intercept)                      Sepal.Length
##                           -0.5694                            0.7985
##                Speciesversicolor                   Speciesvirginica
##                            1.4416                            2.0157
## Sepal.Length:Speciesversicolor   Sepal.Length:Speciesvirginica
##                           -0.4788                           -0.5666
```

```r
iris$preds <- predict(model_species_intercept_slope, newdata=iris)
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species)) +
  geom_point() +
  geom_line(aes(x=Sepal.Length, y=preds, color=Species))+
  facet_grid(. ~ Species) + ggtitle("Different intercepts and slopes")
```




```r
# base vs second model
anova(base_model, model_species_intercept)
```

```
## Analysis of Variance Table
##
## Model 1: Sepal.Width ~ Sepal.Length
## Model 2: Sepal.Width ~ Sepal.Length + Species
##   Res.Df    RSS Df Sum of Sq     F    Pr(>F)
## 1    148 27.916
## 2    146 12.193  2    15.723 94.13 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Ho: adding species info does not improve the model
# Conclusion: F statistic is big, p value < 0.05, we reject Ho
# Therefore, adding species info does improve the model

# base vs third model
anova(model_species_intercept, model_species_intercept_slope)
```

```
## Analysis of Variance Table
##
## Model 1: Sepal.Width ~ Sepal.Length + Species
## Model 2: Sepal.Width ~ Sepal.Length * Species
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1    146 12.193
## 2    144 10.680  2    1.5132 10.201 7.19e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Ho: modelling the petal length independently per species does not improve the model
# Conclusion: F statistic is big, p value < 0.05, we reject Ho

## note: in both cases, using a more complex model allowed to fit the data better
```

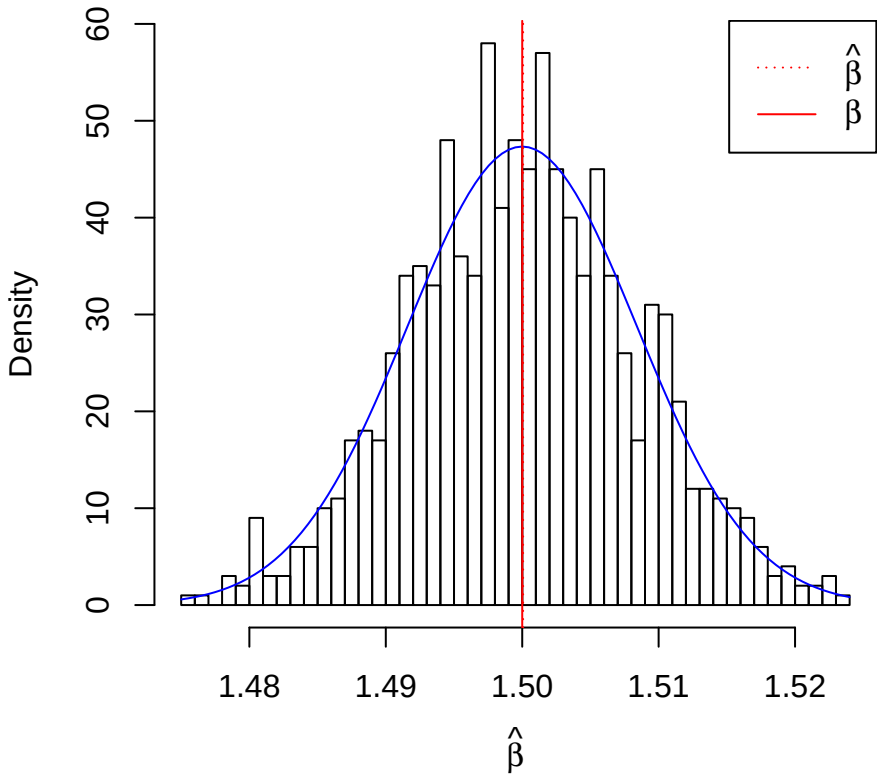
Section 03 - Simulation of the estimates of β

Assume a simple linear model with parameters $\alpha = -0.5, \beta = 1.5, \sigma^2 = 0.7$.

- First simulate a fixed set of $N = 500$ values for x from a normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 20$.
- Then, run the simulation 1000 times. Each time, draw the values of y according to the model specification $y = \alpha + \beta x$, estimate a linear model from the data and store the value of $\hat{\beta}$.
- Finally, create a histogram of the values of $\hat{\beta}$. Indicate the true β and the mean of $\hat{\beta}$ by vertical lines. Also indicate the theoretical distribution of β as a curve.

```r
N <-  500
alpha <- -0.5
beta <- 1.5
sigma.sq <- 0.7
fixed.x <- rnorm(N, sd=sqrt(20))

sim <- function(alpha, beta, sigma.sq, x) {
  # draw y according to the model
  y <- alpha + beta * x + rnorm(length(x), 0, sd = sqrt(sigma.sq))
  m <- lm(y ~ x)
  return(m)
}

# run the simulation 1000 times
beta.hat <- replicate(1e3, coefficients(sim(alpha, beta, sigma.sq, fixed.x))["x"])
hist(beta.hat,freq=FALSE, breaks=50, xlab=expression(hat(beta)),main="")
curve(dnorm(x, beta, sd=sqrt(sigma.sq / (N * var(fixed.x)))), add=TRUE, col="blue")
```

```r
abline(v=beta, col="red")
abline(v=mean(beta.hat), col="red", lty="dotted")
legend("topright", legend=c(expression(hat(beta)), expression(beta)), lty=c(3, 1), col="red")
```



[Optional] Section 04 - Perform a simulation for nested models

Assume two nested linear models:

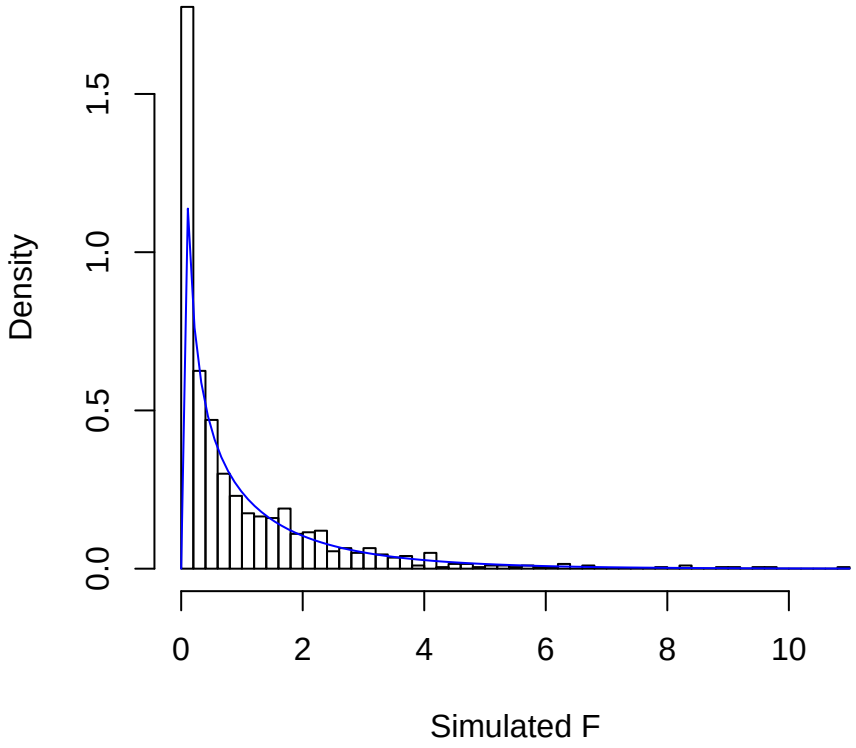
- one reduced model with parameters $\beta_0 = -0.5, \beta_1 = 1.5, \sigma^2 = 0.7$
- full model with parameters $\beta_0 = -0.5, \beta_1 = 1.5, \beta_2 = -0.5, \sigma^2 = 0.7$

1. Simulate a fixed set of $N = 500$ values for x_1 and for x_2 from a normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 20$. Run the simulation 1000 times. In each simulation: * draw the values of y according to the reduced model * fit the reduced and the full model to the simulated data * compute the F statistic of the model comparison

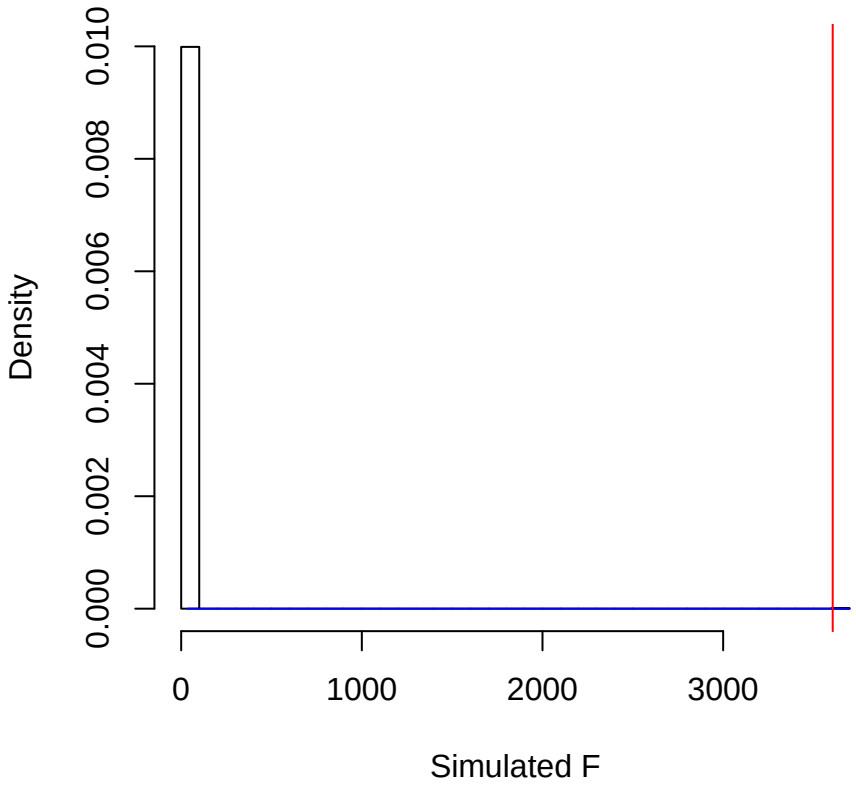
2. Create a histogram of the simulated F statistics. Plot the theoretical distribution of the F statistic on top of the histogram

3. Draw values of y according to the full model (once). Fit the reduced and the full model to the simulated data. Compute the F statistic for the model comparison. Compare the F statistic from the full model to the distribution of F statistics from the reduced model

```r
N <-  500
beta0 <- -0.5
beta1 <- 1.5
beta2 <- -0.5
## make a vector
beta.full <- c(beta0, beta1, beta2)
beta.reduced <- c(beta0, beta1, 0)
sigma.sq <- 0.7

## simulate the fixed x
fixed.x1 <- rnorm(N, sd=sqrt(20))
fixed.x2 <- rnorm(N, sd=sqrt(20))
x = cbind(x1=fixed.x1, x2=fixed.x2)

sim <- function(beta, sigma.sq, x) {
  ## add the intercept
  xx = cbind(intercept=1, x)
  # draw y according to the model
  y <- rnorm(nrow(x), xx %*% beta, sd = sqrt(sigma.sq))
  x = data.frame(y, x)
  full <- lm(y ~ x1 + x2, data=x)
  reduced <- lm(y ~ x1, data=x)
  ## print(anova(reduced, full))
  return(anova(reduced, full)[2,"F"])
}

# run the simulation 1000 times
Fsim <- replicate(1e3, sim(beta.reduced, sigma.sq, x))
hist(Fsim, freq=FALSE, breaks=50, xlab="Simulated F", main="")
curve(df(x, df1=1, df2=N-2), add=TRUE, col="blue")
```




```
## Draw y from the full model
Ffull <- sim(beta.full, sigma.sq, x)
Fsim = c(Fsim, Ffull)
hist(Fsim, freq=FALSE, breaks=50, xlab="Simulated F", main="")
curve(df(x, df1=1, df2=N-2), add=TRUE, col="blue")
abline(v=Ffull, col='red')
```


Density
0.000
0.002
0.004
0.006
0.008
0.010
0
1000
2000
3000
Simulated F